Science of Computer Programming 1 (1982) 235–254
North-Holland Publishing Company

AN EXERCISE IN PROGRAM SYNTHESIS: ALGORITHMS FOR COMPUTING THE TRANSITIVE CLOSURE OF A RELATION

Lothar SCHMITZ
Fachbereich Informatik, Hochschule der Bundeswehr München, Werner-Heisenberg-Weg 39, D-8014 Neubiberg, Fed. Rep. Germany

Communicated by M. Sintzoff
Received November 1981

Abstract. This paper contains the synthesis of several transitive closure algorithms (including Warshall's) from one common high level definition.

For deriving recursion equations Burstall's and Darlington's unfolding- and folding-technique is used. A special effort is made to treat the first step of the syntheses (i.e. finding appropriate recursion arguments) systematically.

1. Introduction

In [5] Darlington has shown how to use the transformation system developed by Burstall and Darlington [1] to synthesize programs from high level specifications. These techniques were (in [6]) and in [4]) successfully applied to the synthesis of several sorting algorithms. In this paper we are trying to do a similar exercise for transitive closure algorithms.

Section 2 contains the problem definition as well as a few lemmata needed for the syntheses. In Section 3 the synthesis technique is explained. The different syntheses (Sections 5–7) are based on the different parametrizations introduced in Section 4. In a final section we summarize what this exercise has revealed to us.

We use the usual notation of informal set theory and additionally the conventions listed below.

the expression	stands for
$\langle x_1, \ldots, x_n \rangle$	ordered n-tuple
$\# M$	cardinality of a set M
$\mathcal{P}(M)$	power set of a set M
$\forall a \leq i \leq b: \ldots$	$\forall i: (i \in \mathbb{N} \wedge a \leq i \leq b \rightarrow \ldots)$
$\exists a \leq i < b: \ldots$	$\exists i: (i \in \mathbb{N} \wedge a \leq i < b \wedge \ldots)$
$x + X$	$\{x\} \cup X$ where $x \notin X$
$\mathbb{N}_0$	$\mathbb{N} \cup \{0\}$

0167-6423/82/0000-0000/$02.75

2. The transitive closure of a relation

Let V be a finite set and $R \subseteq V^2$ a binary relation over V. The elements of V are called vertices, the elements of R edges.

Definition 2.1. The transitive closure R^+ of R is the set of all pairs $\langle x, y\rangle \in V^2$ that are connected by a path in R:

$$\begin{aligned}R^+ =_{\text{def}} \{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}:\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\},\end{aligned}$$

x is the first node, y the last node and $\{v_1, \ldots, v_{m-1}\}$ the set of inner nodes of the path characterized by the node sequence $\langle v_0, \ldots, v_m\rangle$; m is called the length of the path.

Two paths characterized by the sequences $\langle v_0, \ldots, v_m\rangle$ and $\langle v'_0, \ldots, v'_n\rangle$ will be considered equivalent whenever $v_0 = v'_0$ and $v_m = v'_n$.

For program synthesis we shall need several lemmata stating properties of paths. It is convenient to define

Definition 2.2. $path(x, y) \Leftrightarrow_{\text{def}} \langle x, y\rangle \in R^+$.

A first lemma is

Lemma 2.3

$$\begin{aligned}path(x, y) \Leftrightarrow \exists m \in \mathbb{N}: \exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\ &[\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R] \wedge\\ &\forall 0 < i < m: \forall 0 \leqslant j \leqslant m:\\ &[i \neq j \to v_i \neq v_j]],\end{aligned}$$

i.e. for any path p there exists an equivalent path p′ whose inner nodes are touched exactly once; the same holds for the edges of p′.

Proof. "<" obvious. ">" As long as there are inner nodes v that are touched more than once we apply the following construction:

$$first =_{\text{def}} \min(\{i \mid i \in \{1, \ldots, m-1\} \wedge v_i = v\}),$$

$$last =_{\text{def}} \max(\{i \mid i \in \{1, \ldots, m-1\} \wedge v_i = v\}),$$

$$p' =_{\text{def}} \begin{cases} \langle v_{last}, \ldots, v_m \rangle & \text{if } x = v, \\ \langle v_0, \ldots, v_{first} \rangle & \text{if } y = v, \\ \langle v_0, \ldots, v_{first}, v_{last+1}, \ldots, v_m \rangle & \text{otherwise.} \end{cases} \qquad \square$$

The next lemma is an immediate corollary to Lemma 2.3:

Lemma 2.4.

$$path(x, y) \asymp \exists 1 \leqslant m \leqslant \# V: \exists \langle v_0, \ldots, v_m \rangle \in \{x\} \times V^{m-1} \times \{y\}:$$
$$\forall 0 \leqslant i < m: [\langle v_i, v_{i+1} \rangle \in R],$$

i.e. for any path there exists an equivalent path whose length is less than or equal to $\# V$.

The following obvious lemma will facilitate the decomposition of paths:

Lemma 2.5.

$$path(x, y) \asymp \exists a, b \in \mathbb{N}_0: [a < b \wedge$$
$$\exists \langle v_a, \ldots, v_b \rangle \in \{x\} \times V^{b-a-1} \times \{y\}:$$
$$\forall a \leqslant i < b: [\langle v_i, v_{i+1} \rangle \in R]].$$

The judicious choice of properties as above depends on experience and good judgement: this issue is not tackled in the present paper.

3. The synthesis technique

Our aim is to derive recursion equations from a given specification. This is done in a few steps. We illustrate our method by deriving a recursive program that computes the inner product of two vectors, $a = (a_1, a_2, a_3)$ and $b = (b_1, b_2, b_3)$. The value IP we want to compute is specified by the equation

$$IP =_{\text{def}} \sum_{i=1}^{3} a_i \cdot b_i. \tag{3.1}$$

3.1. *Parametrization*

Let S be a specification and e be an entity that occurs unbound in S. Then by functional abstraction we obtain $\lambda e.S$. We will write S_e instead of $\lambda e.S$ and will speak of a parametrization (or generalization) of S. In order to guarantee termination of the program being developed, e has to be an element of a well-founded set $(M, <)$ (i.e. there is no infinite decreasing sequence $m_1 > m_2 > m_3 > \cdots$ in the partially ordered set M). In the specification (3.1) i does not occur unbound but the lower bound, 1, does. So we can use the following definitions as an illustration:

$$
\begin{aligned}
&e =_{\mathrm{def}} 1,\\
&M =_{\mathrm{def}} \{m \in \mathbb{N} \mid 1 \leq m \leq 3\}, \qquad\qquad (3.2)\\
&\prec \; =_{\mathrm{def}} \; > \cap M^2, \quad \text{e.g. } 3 \prec 2 \text{ since } 3 > 2 \text{ and } (3,2) \in M^2.
\end{aligned}
$$

We get, using 1 as a symbol of a bound variable,

$$IP_1 = \lambda 1.IP = \lambda 1. \sum_{i=1}^{3} a_i \cdot b_i. \tag{3.3}$$

3.2. Termination equations

For every minimal element min of M we have to specify the value val of $S_e(\text{min})$. The only minimal element of the well-founded set $(M, \prec)$ in (3.2) in 3. So by instantiating (3.3) we obtain

$$IP_1(3) = \left(\lambda 1. \sum_{i=1}^{3} a_i \cdot b_i\right)(3) = \sum_{i=3}^{3} a_i \cdot b_i = a_3 \cdot b_3. \tag{3.4}$$

3.3. Recursion equations

For every non-minimal element x of M we have to find recursion equations of the following form:

$$S_e(x) = f(S_e(x_1), \ldots, S_e(x_n)) \tag{3.5}$$

where $\forall 1 \leq i \leq n: x_i \prec x$ and f is a function we know how to compute. Thus, (3.5) reduces the problem of computing $S_e(x)$ to the problems of computing $S_e(x_i)$ for some x_i smaller than x.

How do we find equations of the form (3.5)? We manipulate the right-hand side of $S_e(x)$ trying to isolate right-hand sides of $S_e(x_i)$ (for $x_i \prec x$) that can be replaced by their corresponding left-hand sides (this strategy is introduced as "forced folding" in [1]).

For our simple example we get (remember $x+1 \prec x$)

$$
\begin{aligned}
IP_1(x) &= \sum_{i=x}^{3} a_i \cdot b_i\\
&= a_x \cdot b_x + \sum_{i=x+1}^{3} a_i \cdot b_i\\
&= a_x \cdot b_x + IP_1(x+1).
\end{aligned} \tag{3.6}
$$

3.4. Recursive programs

$$S_e(e) = (\lambda e.S)(e) = S \tag{3.7}$$

holds for any parametrization of any specification S. So, putting things together and fixing a direction of evaluation, we obtain recursive programs of the following form:

$$
\begin{aligned}
&S \Leftarrow S_e(e), \\
&S_e(min) \Leftarrow val \quad \text{for every minimal } min \in M, \\
&S_e(x) \Leftarrow f(S_e(x_1), \ldots, S_e(x_n)) \quad \text{for every non-minimal } x \in M.
\end{aligned} \tag{3.8}
$$

We have derived the following recursive program:

$$
\begin{aligned}
&IP \Leftarrow IP_1(1), \\
&IP_1(3) \Leftarrow a_3 \cdot b_3, \\
&IP_1(x) \Leftarrow a_x \cdot b_x + IP_1(x+1).
\end{aligned} \tag{3.9}
$$

3.5. Standard well-founded sets

In this paper e will be either a natural number or a finite set. For $e \in \mathbb{N}$ we use

$$M =_{\text{def}} \mathbb{N} \quad \text{and} \quad \prec =_{\text{def}} <. \tag{3.10}$$

The only minimal element of this well-founded set is $min = 1$. Elements smaller than $x \neq 1$ are e.g.

$$y \quad \text{if } x = y + 1, \tag{3.11}$$

$$y \quad \text{if } x = 2 * y. \tag{3.12}$$

If e is a finite set we use

$$M =_{\text{def}} \mathcal{P}(e) \quad \text{and} \quad \prec =_{\text{def}} \subset. \tag{3.13}$$

The only minimal element of this well-founded set is $\min = \emptyset$. Elements smaller than $X \neq \emptyset$ are e.g.

$$\{x\} \text{ and } Y \quad \text{if } X = x + Y, \tag{3.14}$$

$$X_1 \text{ and } X_2 \quad \text{if } X = X_1 \cup X_2 \text{ and } X_1 \neq \emptyset \neq X_2. \tag{3.15}$$

4. Different parametrizations

We consider the entities

$$x, y, V, m, \mathbb{N}, v_0, \ldots, v_m, i, R$$

occurring in Definition 2.1.

Of these only V, $\mathbb{N}$ and R occur unbound. $(\mathcal{P}(\mathbb{N}), \subset)$ is not a well-founded set because $\langle\{n \cdot 2^i \mid n \in \mathbb{N}\}\rangle_{i \in \mathbb{N}}$ is an infinite decreasing sequence. So we are left with the entities V and R.

V occurs in V^2 and V^{m-1}. Parametrization relative to the occurrences of V in V^2 seems to result in algorithms that exhaust the definition with a kind of British Museum method. So we use the occurrences of V in V^{m-1} for parametrization and get

$$\begin{aligned} R_V^+(I) = \{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times I^{m-1} \times \{y\}:\\ &\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\}. \end{aligned} \tag{4.1}$$

(Instead of "I" we could have used any other letter. We chose "I" to indicate that I^{m-1} is the domain the sequence of inner nodes is taken from.)

Using R for parametrization we obtain

$$\begin{aligned} R_R^+(K) = \{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in K]\}. \end{aligned} \tag{4.2}$$

Application of Lemma 2.4 to Definition 2.1 yields another definition for R^+:

Definition 4.3.

$$\begin{aligned} R^+ =_{\text{def}} \{\langle x, y\rangle \in V^2 \mid \exists 1 \leq m \leq \# V:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\}. \end{aligned}$$

This gives us the opportunity to use the natural number $\# V$ for parametrization:

$$\begin{aligned} R_{\# V}^+(l) = \{\langle x, y\rangle \in V^2 \mid \exists 1 \leq m \leq l:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\}. \end{aligned} \tag{4.4}$$

(We could have used the lower bound, 1, equally well for parametrization but the corresponding well-founded set would have looked as clumsy as the one in (3.2).)

Comment. (4.1), (4.2) and (4.4) may be interpreted as follows:

(i) $\langle x, y\rangle \in R_V^+(I)$ iff there exists a path from x to y whose inner nodes are taken from I;

(ii) $\langle x, y\rangle \in R_R^+(K)$ iff there exists a path from x to y whose edges belong to K;

(iii) $\langle x, y\rangle \in R_{\# V}^+(l)$ iff there exists a path from x to y whose length m is less than or equal to l.

5. Syntheses based on the parametrization R^+_{*V}

We are using natural numbers (3.10). So the only minimal element is 1. Instantiating (4.4), we obtain

$$\begin{aligned}R^+_{*V}(1) &= \{\langle x, y\rangle \in V^2 \mid \exists 1 \leqslant m \leqslant 1:\\ &\qquad \exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\qquad \forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}\\ &= \{\langle x, y\rangle \in V^2 \mid \exists\langle v_0, v_1\rangle \in \{x\} \times \{y\}: [\langle v_0, v_1\rangle \in R]\}\\ &= \{\langle x, y\rangle \in V^2 \mid \langle x, y\rangle \in R\}\\ &= R.\end{aligned} \tag{5.1}$$

This is the termination equation for the following two syntheses.

5.1. Use of ordering (3.11)

Let $l \geqslant 1$. We start with the right-hand side belonging to

$$R^+_{*V}(l+1), \tag{5.2}$$

namely

$$\begin{aligned}\{\langle x, y\rangle \in V^2 \mid \exists 1 \leqslant m \leqslant l+1:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}\end{aligned} \tag{5.3}$$

and try to isolate instances of the right-hand side belonging to $R^+_{*V}(l)$.

The quantification "$\exists 1 \leqslant m \leqslant l+1: \ldots$" may be split into "$\exists 1 \leqslant m \leqslant l: \ldots \vee \exists 2 \leqslant m \leqslant l+1: \ldots$" yielding

$$\begin{aligned}\{\langle x, y\rangle \in V^2 \mid \exists 1 \leqslant m \leqslant l:&\\ &\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\\ &\vee \exists 2 \leqslant m \leqslant l+1:\\ &\quad\exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\quad\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}.\end{aligned} \tag{5.4}$$

Folding the first three lines results in

$$\begin{aligned}&R^+_{*V}(l) \cup\\ &\{\langle x, y\rangle \in V^2 \mid \exists 2 \leqslant m \leqslant l+1:\\ &\qquad \exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:\\ &\qquad \forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}.\end{aligned} \tag{5.5}$$

To prepare another fold we introduce a node z and identify v_{m-1} with z. Because of $m \geq 2$ the inner node v_{m-1} exists.

$$\begin{aligned}
&R^+_{\# V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists z \in V: \exists 2 \leq m \leq l+1: \\
&\qquad \exists\langle v_0, \dots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\
&\qquad \forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R] \wedge z = v_{m-1}\}.
\end{aligned} \tag{5.6}$$

"$\forall 0 \leq i < m: \langle v_i, v_{i+1}\rangle \in R$" is equivalent to
"$\forall 0 \leq i < m-1: \langle v_i, v_{i+1}\rangle \in R \wedge \langle z, y\rangle \in R$", given the context.

Now v_m may be dropped from the quantifier as it does not occur anywhere else. Pulling out "$\langle z, y\rangle \in R$" as far as possible, we obtain

$$\begin{aligned}
&R^+_{\# V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists z \in V: [\langle z, y\rangle \in R \wedge \\
&\qquad \exists 2 \leq m \leq l+1: \exists\langle v_0, \dots, v_{m-1}\rangle \in \{x\} \times V^{m-2} \times \{z\}: \\
&\qquad \forall 0 \leq i < m-1: [\langle v_i, v_{i+1}\rangle \in R]]\}.
\end{aligned} \tag{5.7}$$

After the index transformation $m' =_{\text{def}} m-1$ the last two lines read:

$$\text{"}\exists 1 \leq m' \leq l: \exists\langle v_0, \dots, v_{m'}\rangle \in \{x\} \times V^{m'-1} \times \{z\}:$$

$$\forall 0 \leq i < m-1: [\langle v_i, v_{i+1}\rangle \in R]\text{"}$$

which is equivalent to "$\langle x, z\rangle \in R^+_{\# V}(l)$".

This completes the derivation of the recursion equation

$$\begin{aligned}
R^+_{\# V}(l+1) = R^+_{\# V}(l) \cup \\
\{\langle x, y\rangle \in V^2 \mid \exists z \in V: [\langle z, y\rangle \in R \\
\wedge \langle x, z\rangle \in R^+_{\# V}(l)]\}
\end{aligned} \tag{5.8}$$

and we can write down our first recursive program $TC_{\# V,1}$ that computes the transitive closure R^+ of a given relation R.

$TC_{\# V,1}$:

$$R^+ \Leftarrow R^+_{\# V}(\# V)$$

$$R^+_{\# V}(1) \Leftarrow R$$

$$R^+_{\# V}(l+1) \Leftarrow R^+_{\# V}(l) \cup \{\langle x, y\rangle \in V^2 \mid \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle x, z\rangle \in R^+_{\# V}(l)\}$$

The person or machine that is to execute this program has to know how to enumerate the elements $\langle x, y\rangle$ of the Cartesian product V^2 and how to evaluate the quantification $\exists z \in V$. If he cannot do this we have to refine the program for him.

The part that has to be refined is the computation of the set

$$H(l) =_{\text{def}} \{\langle x, y\rangle \in V^2 \mid \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle x, z\rangle \in R^+_{\# V}(l)]\}$$
$$= \{\langle x, y\rangle \mid x \in V \wedge y \in V \wedge \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle x, z\rangle \in R^+_{\# V}(l)]\}. \tag{5.9}$$

We parameterize the three occurrences of V one after another. V being a finite set we can use (3.13). The first parametrization yields

$$H_1(l, V_1) = \{\langle x, y\rangle \mid x \in V_1 \wedge y \in V \wedge \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle x, z\rangle \in R^+_V(l)]\}. \tag{5.10}$$

Instantiating with $V_1 = \emptyset$ we obtain the termination equation

$$H_1(l, \emptyset) = \{\langle x, y\rangle \mid x \in \emptyset \ldots\}$$
$$= \emptyset. \tag{5.11}$$

Now using (3.14) we get

$$H_1(l, v_1 + V_1) = \{\langle x, y\rangle \mid x \in v_1 + V_1 \wedge y \in V \wedge \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle x, z\rangle \in R^+_{\# V}(l)]\}$$
$$= (\text{splitting up "}x \in v_1 + V_1\text{" into "}x = v_1 \vee x \in V_1\text{"})$$
$$\{\langle x, y\rangle \mid x = v_1 \wedge y \in V \wedge \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle v_1, z\rangle \in R^+_V(l)]\}$$
$$\cup H_1(l, V_1)$$
$$= H_2(l, v_1, V) \cup H_1(l, V_1) \tag{5.12}$$

where

$$H_2(l, v_1, V_2) =_{\text{def}} \{\langle x, y\rangle \mid x = v_1 \wedge y \in V_2 \wedge \exists z \in V:$$
$$[\langle z, y\rangle \in R \wedge \langle v_1, z\rangle \in R^+_{\# V}(l)]\}. \tag{5.13}$$

Proceeding in the same straightforward manner we obtain equations for H_2 and H_3 (which is introduced in the recursion equation for H_2) and eventually we arrive at the following program:

$TC_{\# V,2}$:

$R^+ \Leftarrow R^+_{\# V}(\# V)$

$R^+_{\# V}(1) \Leftarrow R$

$R^+_{\# V}(l+1) \Leftarrow R^+_{\# V}(l) \cup H_1(l, V)$

$$H_1(l, \emptyset) \Leftarrow \emptyset$$

$$H_1(l, v_1 + V_1) \Leftarrow H_1(l, V_1) \cup H_2(l, v_1, V)$$

$$H_2(l, v_1, \emptyset) \Leftarrow \emptyset$$

$$H_2(l, v_1, v_2 + V_2) \Leftarrow H_2(l, v_1, V_2) \cup H_3(l, v_1, v_2, V)$$

$$H_3(l, v_1, v_2, \emptyset) \Leftarrow \emptyset$$

$$H_3(l, v_1, v_2, v_3 + V_3) \Leftarrow H_3(l, v_1, v_2, V_3) \cup \{\langle v_1, v_2\rangle \mid \langle v_3, v_2\rangle \in R \wedge \langle v_1, v_3\rangle \in R^+_{\# V}(l)\}$$

This program is certainly less readable than $TC_{\# V,1}$. But if we write it down in a more conventional notation using loops (hopefully the introduction of programming language constructs will be accomplished systematically with the aid of a system like [2, 3]) we recognize that our program computes the transitive closure by repetitively "multiplying" a boolean matrix with itself (see e.g. [8, section 2.7]):

$TC_{\# V,3}$:

$R^+_{\# V}(l) \Leftarrow R$

for l **from** 2 **to** $\# V$

  $R^+_{\# V}(l) \Leftarrow R^+_{\# V}(l-1)$

  for $v_1 \in V$

    for $v_2 \in V$

      for $v_3 \in V$

        if $\langle v_1, v_3\rangle \in R^+_{\# V}(l-1) \wedge \langle v_3, v_2\rangle \in R$

        then $R^+_{\# V}(l) \Leftarrow R^+_{\# V}(l) \cup \{\langle v_1, v_2\rangle\}$ ⅃⅃⅃⅃

$R^+ \Leftarrow R^+_{\# V}(\# V)$

The essence of the algorithm is most clearly seen in $TC_{\# V,1}$. So in future syntheses we will omit straightforward and tedious refinements.

5.2. *Use of ordering (3.12)*

Let again be $l \geq 1$. This time we start with the right-hand side belonging to

$$R^+_{\# V}(2 * l), \tag{5.14}$$

i.e. with

$$\{\langle x, y\rangle \in V^2 \mid \exists 1 \leq m \leq 2 * l: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\} \tag{5.15}$$

and again try to isolate instances of the right-hand side belonging to $R^+_{\# V}(i)$.

We split "$\exists 1 \leqslant m \leqslant 2*l: \ldots$" into "$\exists 1 \leqslant m \leqslant l: \ldots \vee \exists l+1 \leqslant m \leqslant 2*l: \ldots$" to obtain

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \exists 1 \leqslant m \leqslant l: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\
&\qquad \forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R] \\
&\qquad \vee \exists l+1 \leqslant m \leqslant 2*l: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\
&\qquad \forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}
\end{aligned} \tag{5.16}$$

Folding the first two lines of (5.16) yields

$$\begin{aligned}
&R^+_{\#V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists l+1 \leqslant m \leqslant 2*l: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\
&\forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]\}.
\end{aligned} \tag{5.17}$$

To find more opportunities to fold we suitably split up $\langle v_0, \ldots, v_m\rangle$ into two parts, $\langle v_0, \ldots, v_t\rangle$ and $\langle v_t, \ldots, v_m\rangle$, where $m - l \leqslant t \leqslant l$. For m fulfilling $l+1 \leqslant m \leqslant 2*l$ such a t exists, e.g. $t =_{\mathrm{def}} m \operatorname{div} 2$. We introduce a node z and identify it with v_t like we did with v_{m-1} in (5.6):

$$\begin{aligned}
&R^+_{\#V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists z \in V: \exists l+1 \leqslant m \leqslant 2*l: \exists m - l \leqslant t \leqslant l: \\
&\qquad \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\
&\qquad [z = v_t \wedge \forall 0 \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]]\}.
\end{aligned} \tag{5.18}$$

Next we separate $\langle v_0, \ldots, v_t\rangle$ from $\langle v_t, \ldots, v_m\rangle$ to get

$$\begin{aligned}
&R^+_{\#V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists z \in V: \exists l+1 \leqslant m \leqslant 2*l: \exists m - l \leqslant t \leqslant l: \\
&\qquad [\exists \langle v_0, \ldots, v_t\rangle \in \{x\} \times V^{t-1} \times \{z\}: \\
&\qquad \forall 0 \leqslant i < t: [\langle v_i, v_{i+1}\rangle \in R] \\
&\qquad \wedge \exists \langle v_t, \ldots, v_m\rangle \in \{z\} \times V^{m-t-1} \times \{y\}: \\
&\qquad \forall t \leqslant i < m: [\langle v_i, v_{i+1}\rangle \in R]]\}.
\end{aligned} \tag{5.19}$$

Using Lemma 2.5 and folding twice finishes the derivation of the recursion equation

$$\begin{aligned}
R^+_{\#V}(2*l) = R^+_{\#V}(l) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \exists z \in V: \\
&\qquad [\langle x, z\rangle \in R^+_{\#V}(l) \wedge \langle z, y\rangle \in R^+_{\#V}(l)]\}.
\end{aligned} \tag{5.20}$$

Now $\# V$ usually is not a power of 2 and we have to take care of odd arguments somehow. Lemma 2.4 implies

$$\forall n \in \mathbb{N}: [R^+_{\#V}(\# V + n) = R^+_{\#V}(\# V)]. \tag{5.21}$$

Let i_0 be the smallest i fulfilling $2^i \geq \# V$. As $R^+_{\# V}(\# V) = R^+$ we have

$$R^+_{\# V}(2^{i_0}) = R^+_{\# V}(\# V) = R^+. \tag{5.22}$$

So when starting with 2^{i_0} only even arguments will occur.

We obtain the following recursive program:

$TC_{\# V,4}$:

$$R^+ \Leftarrow R^+_{\# V}(2^{i_0}), \text{ where } i_0 = \min\{i \mid 2^i \geq \# V\}$$

$$R^+_{\# V}(1) \Leftarrow R$$

$$R^+_{\# V}(2 * l) \Leftarrow R^+_{\# V}(l) \cup \{\langle x, y\rangle \in V^2 \mid \exists z \in V: [\langle x, z\rangle \in R^+_{\# V}(l) \wedge \langle z, y\rangle \in R_{\# V}(l)]\}.$$

This program uses bisection and is therefore more efficient than $TC_{\# V,1}$.

6. Synthesis based on the parametrization $R^+_V(I)$

We again use (3.13). So we obtain as termination-equation

$$\begin{aligned}
R^+_V(\emptyset) &= \{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \\
&\qquad \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times \emptyset^{m-1} \times \{y\}: \\
&\qquad \forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\} \\
&= (\text{for } m \neq 1, \text{ the Cartesian product is empty}) \\
&\quad \{\langle x, y\rangle \in V^2 \mid \exists \langle v_0, v_1\rangle \in \{x\} \times \{y\}: \\
&\qquad [\langle v_0, v_1\rangle \in R]\} \\
&= \{\langle x, y\rangle \in V^2 \mid \langle x, y\rangle \in R\} \\
&= R.
\end{aligned} \tag{6.1}$$

Using (3.14) we start with the right-hand side of

$$R^+_V(b + B), \tag{6.2}$$

namely

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \\
&\exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b + B)^{m-1} \times \{y\}: \\
&\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in R]\}
\end{aligned} \tag{6.3}$$

and aim to isolate instances of the right-hand side of $R^+_V(B)$.

If b does not occur as an inner node, (6.3) immediately reduces to $R^+_V(B)$.
So by adding the tautology

$$\text{“}b \in \{v_1, \ldots, v_{m-1}\} \vee b \notin \{v_1, \ldots, v_{m-1}\}\text{”}$$

we introduce a case analysis:

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \\
&\quad \exists m \in \mathbb{N}\colon \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b+B)^{m-1} \times \{y\}\colon \\
&\quad [b \in \{v_1, \ldots, v_{m-1}\rangle \wedge \forall 0 \leq i < m\colon [\langle v_i, v_{i+1}\rangle \in R]] \\
&\vee \exists m \in \mathbb{N}\colon \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b+B)^{m-1} \times \{y\}\colon \\
&\quad [b \notin \{v_1, \ldots, v_{m-1}\} \wedge \forall 0 \leq i < m\colon [\langle v_i, v_{i+1}\rangle \in R]]\}.
\end{aligned} \tag{6.4}$$

“$\langle v_1, \ldots, v_{m-1}\rangle \in (b+B)^{m-1}$” and “$b \notin \{v_1, \ldots, v_{m-1}\}$” imply “$\langle v_1, \ldots, v_{m-1}\rangle \in B^{n-1}$”. Thus the last two lines of (6.4) may be folded, yielding

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \\
&\exists m \in \mathbb{N}\colon \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b+B)^{m-1} \times \{y\}\colon \\
&[b \in \{v_1, \ldots, v_{m-1}\} \wedge \forall 0 \leq i < m\colon [\langle v_i, v_{i+1}\rangle \in R]]\} \\
&\cup R^+_V(B).
\end{aligned} \tag{6.5}$$

Applying Lemma 2.3 we replace “$b \in \{v_1, \ldots, v_{m-1}\}$” by “$\exists_1 1 \leq i_b \leq m-1\colon [b = v_{i_b}]$”.
Pulling out “$\exists_1 1 \leq i_b \leq m-1$:” we obtain

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}\colon \exists_1 1 \leq i_b \leq m-1\colon \\
&\exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b+B)^{m-1} \times \{y\}\colon \\
&[\forall 0 \leq i < m\colon [\langle v_i, v_{i+1}\rangle \in R] \wedge b = v_{i_b}]\} \\
&\cup R^+_V(B).
\end{aligned} \tag{6.6}$$

As

$$\text{“}\exists_1 1 \leq i_b \leq m-1\colon \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times (b+B)^{m-1} \times \{y\}\colon [b = v_{i_b}]\text{”}$$

is equivalent to

$$\text{“}\exists 1 \leq i_b \leq m-1\colon \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times B^{i_b-1} \times \{b\} \times B^{m-i_b-1} \times \{y\}\text{”}$$

we can split “$\langle v_0, \ldots, v_m\rangle$” into “$\langle v_0, \ldots, v_{i_b}\rangle$” and “$\langle v_{i_b}, \ldots, v_m\rangle$”:

$$\begin{aligned}
&\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}\colon \exists 1 \leq i_b \leq m-1\colon \\
&[\exists \langle v_0, \ldots, v_{i_b}\rangle \in \{x\} \times B^{i_b-1} \times \{b\}\colon \\
&\forall 0 \leq i < i_b\colon [\langle v_i, v_{i+1}\rangle \in R] \wedge
\end{aligned}$$

$$\exists \langle v_{i_b}, \ldots, v_m \rangle \in \{b\} \times B^{m-i_b-1} \times \{y\}:$$
$$\forall i_b \leq i < m: [\langle v_i, v_{i+1} \rangle \in R]]\}$$
$$\cup R_V^+(B). \tag{6.7}$$

Using Lemma 2.5 the first three lines of (6.7) may be folded yielding "$\langle x, b\rangle \in R_V^+(B)$". In the same way we obtain "$\langle b, y\rangle \in R_V^+(B)$" from the first, third and fourth line.

Altogether we have shown

$$R_V^+(b+B) = R_V^+(B) \cup \{\langle x, y\rangle \in V^2 \mid \langle x, b\rangle \in R_V^+(B) \wedge \langle b, y\rangle \in R_V^+(B)\}. \tag{6.8}$$

This completes the derivation of the recursive program

$TC_{V,1}$:

$$R^+ \Leftarrow R_V^+(V)$$
$$R_V^+(\emptyset) \Leftarrow R$$
$$R_V^+(b+B) \Leftarrow R_V^+(B) \cup \{\langle x, y\rangle \in V^2 \mid \langle x, b\rangle \in R_V^+(B) \wedge \langle b, y\rangle \in R_V^+(B)\}$$

By the following reasoning we can slightly optimize program $TC_{V,1}$:

For $b = x$ the expression

$$\text{"}\langle x, y\rangle \in R_V^+(B) \vee \langle x, b\rangle \in R_V^+(B) \wedge \langle b, y\rangle \in R_V^+(B)\text{"}$$

becomes

$$\text{"}\langle x, y\rangle \in R_V^+(B) \vee \langle x, x\rangle \in R_V^+(B) \wedge \langle x, y\rangle \in R_V^+(B)\text{"}$$

which is equivalent to "$\langle x, y\rangle \in R_V^+(B)$".

A similar argument holds for $b = y$. So in $TC_{V,1}$ the set

$$\text{"}\{\langle x, y\rangle \in V^2 \mid \langle x, b\rangle \in R_V^+(B) \wedge \langle b, y\rangle \in R_V^+(B)\}\text{"}$$

may be rewritten as

$$\text{"}\{\langle x, y\rangle \in (V\backslash\{b\})^2 \mid \langle x, b\rangle \in R_V^+(B) \wedge \langle b, y\rangle \in R_V^+(B)\}\text{"}.$$

We may translate this into a more conventional notation and replace recursion by iteration (for a formal treatment we again set our hopes in powerful program transformations):

$TC_{V,2}$:

$R^+ \Leftarrow R$

for $b \in V$

```
⌈for x ∈ V\{b}
  ⌈for y ∈ V\{b}
    ⌈if ⟨x, b⟩ ∈ R⁺ ∧ ⟨b, y⟩ ∈ R⁺
     then R⁺ ⇐ R⁺ ∪ {⟨x, y⟩}⌋⌋⌋
```

We recognize Warshall's well-known algorithm [10].

We can further improve its efficiency by converting

$$\{\langle x, y\rangle \in (V\backslash\{b\})^2 \mid \langle x, b\rangle \in R^+_V(B) \wedge \langle b, y\rangle \in R^+_V(B)\} \tag{6.9}$$

into a Cartesian product:

$$\{x \in V\backslash\{b\} \mid \langle x, b\rangle \in R^+_V(B)\} \times \{y \in V\backslash\{b\} \mid \langle b, y\rangle \in R^+_V(B)\}. \tag{6.10}$$

The following program is based on this transformation:

```
TC_{V,3}:
 R⁺ ⇐ R
 for b ∈ V
   ⌈Y ⇐ ∅
    for y ∈ V\{b}
      ⌈if ⟨b, y⟩ ∈ R⁺
       then Y ⇐ Y ∪ {y}⌋
    for x ∈ V\{b}
      ⌈if ⟨x, b⟩ ∈ R⁺
        then for y ∈ Y
               ⌈R⁺ ⇐ R⁺ ∪ {⟨x, y⟩}⌋⌋⌋
```

Notice that both component sets of the Cartesian product are computed only once by $TC_{V,3}$.

7. Synthesis based on the parametrization $R^+_R(K)$

We are using (3.13) again. The termination-equation reads

$$\begin{aligned} R^+_R(\emptyset) = \{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}:\; & \exists\langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\ & \forall 0 \le i < m: [\langle v_i, v_{i+1}\rangle \in \emptyset]\} \\ = \emptyset. & \end{aligned} \tag{7.1}$$

(3.14) causes us to start with the right-hand side of

$$R_R^+(\langle k_1, k_2\rangle + K) \tag{7.2}$$

i.e. with

$$\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:$$
$$\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K]\} \tag{7.3}$$

trying to isolate occurrences of the right-hand side of $R_R^+(K)$.

If the edge $\langle k_1, k_2\rangle$ does not occur on every path from x to y, then (7.3) immediately reduces to $R_R^+(K)$. This is determined by including a tautology:

$$\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:$$
$$[\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge$$
$$(\exists 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle] \vee$$
$$\neg\exists 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle])]\}.$$

"$\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K]$" and "$\neg\exists 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle]$" together imply "$\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in K]$". So the second case reduces to $R_R^+(K)$.

In the first case Lemma 2.3 strengthens "$\exists 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle]$" yielding "$\exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle]$". We obtain

$$\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:$$
$$[\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge$$
$$\exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle]]\}$$
$$\cup R_R^+(K). \tag{7.4}$$

Regarding m and i we distinguish four cases by adding the tautology

"$m = 1 \vee m \neq 1 \wedge (i = 0 \vee i = m-1 \vee 0 < i < m-1)$".

Remark. The four cases convey the following intuitive meaning (all paths being mentioned leading from x to y):

(i) there is a path consisting of the edge $\langle k_1, k_2\rangle$;

(ii) $\langle k_1, k_2\rangle$ is the first edge of a path;

(iii) $\langle k_1, k_2\rangle$ is the last edge of a path;

(iv) $\langle k_1, k_2\rangle$ is neither the first nor the last edge of a path.

Now including the tautology into the scope of the quantification "$\exists_1 0 \leq i < m$" gives us

$$\{\langle x, y\rangle \in V^2 \mid \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}:$$
$$[\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge$$

$$\exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle \wedge$$
$$(m = 1 \vee m \neq 1 \wedge (i = 0 \vee i = m-1 \vee 0 < i < m-1)]]\}$$
$$\cup R_R^+(K). \tag{7.5}$$

Using distributivity we eventually get

$$R_R^+(K) \cup \{\langle x, y\rangle \in V^2 |$$

$$\text{(i)} \begin{cases} \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\ [\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge \\ \exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle \wedge m = 1]] \end{cases}$$

$$\vee$$

$$\text{(ii)} \begin{cases} \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\ [\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge \\ \exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle \wedge m \neq 1 \wedge i = 0]] \end{cases}$$

$$\vee$$

$$\text{(iii)} \begin{cases} \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\ [\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge \\ \exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle \wedge m \neq 1 \wedge i = m-1]] \end{cases}$$

$$\vee$$

$$\text{(iv)} \begin{cases} \exists m \in \mathbb{N}: \exists \langle v_0, \ldots, v_m\rangle \in \{x\} \times V^{m-1} \times \{y\}: \\ [\forall 0 \leq i < m: [\langle v_i, v_{i+1}\rangle \in \langle k_1, k_2\rangle + K] \wedge \\ \exists_1 0 \leq i < m: [\langle v_i, v_{i+1}\rangle = \langle k_1, k_2\rangle \wedge m \neq 1 \wedge 0 < i < m-1]]\}. \end{cases} \tag{7.6}$$

Part (i) immediately reduces to

"$\langle x, y\rangle = \langle k_1, k_2\rangle$" or "$x = k_1 \wedge y = k_2$".

Part (ii) is equivalent to

"$x = k_1 \wedge$
$\exists m \in \mathbb{N} \setminus \{1\}: \exists \langle v_1, \ldots, v_m\rangle \in \{k_2\} \times V^{m-2} \times \{y\}:$
$\forall 1 \leq i < m: [\langle v_i, v_{i+1}\rangle \in K]$"

which using Lemma 2.5 may be folded to yield

"$x = k_1 \wedge \langle k_2, y\rangle \in R_R^+(K)$".

In the same way part (iii) reduces to

$$\text{“}\langle x, k_1\rangle \in R_R^+(K) \wedge y = k_2\text{”}$$

and part (iv) to

$$\text{“}\langle x, k_1\rangle \in R_R^+(K) \wedge \langle k_2, y\rangle \in R_R^+(K)\text{”}.$$

Thus we have obtained

$$\begin{aligned}
&R_R^+(K) \cup \\
&\{\langle x, y\rangle \in V^2 \mid \\
&\quad x = k_1 \wedge y = k_2 \\
&\quad \vee \\
&\quad x = k_1 \wedge \langle k_2, y\rangle \in R_R^+(K) \\
&\quad \vee \\
&\quad \langle x, k_1\rangle \in R_R^+(K) \wedge y = k_2 \\
&\quad \vee \\
&\quad \langle x, k_1\rangle \in R_R^+(K) \wedge \langle k_2, y\rangle \in R_R^+(K)\}
\end{aligned} \tag{7.7}$$

which is equal to

$$\begin{aligned}
&R_R^+(K) \cup \\
&\{\langle x, y\rangle \in V^2 \mid (x = k_1 \vee \langle x, k_1\rangle \in R_R^+(K)) \\
&\qquad\qquad \wedge (y = k_2 \vee \langle k_2, y\rangle \in R_R^+(K))\}.
\end{aligned} \tag{7.8}$$

Like in (6.10) we convert the second part of (7.8) into a Cartesian product

$$\begin{aligned}
&R_R^+(K) \cup \\
&\{x \in V \mid x = k_1 \vee \langle x, k_1\rangle \in R_R^+(K)\} \times \\
&\{y \in V \mid y = k_2 \vee \langle k_2, y\rangle \in R_R^+(K)\}.
\end{aligned} \tag{7.9}$$

This completes the derivation of the program:

$$\begin{aligned}
&TC_{R,1}: \\
&\quad R^+ \Leftarrow R_R^+(R) \\
&\quad R_R^+(\emptyset) \Leftarrow \emptyset \\
&\quad R_R^+(\langle k_1, k_2\rangle + K) \Leftarrow R^+(K) \cup \\
&\qquad\qquad\qquad \{x \in V \mid x = k_1 \vee \langle x, k_1\rangle \in R_R^+(K)\} \times \\
&\qquad\qquad\qquad \{y \in V \mid y = k_2 \vee \langle k_2, y\rangle \in R_R^+(K)\}
\end{aligned}$$

Hopefully the following program in a more conventional notation may be deduced from $TC_{R,1}$:

$TC_{R,2}$:

$R^+ \Leftarrow \emptyset$

for $\langle k_1, k_2 \rangle \in R$

$\lceil X \Leftarrow \{k_1\}$

$Y \Leftarrow \{k_2\}$

for $k \in V$

if $\langle k, k_1 \rangle \in R^+$ **then** $X \Leftarrow X \cup \{k\}$

if $\langle k_2, k \rangle \in R^+$ **then** $Y \Leftarrow Y \cup \{k\}$

for $x \in X$

$\lceil$**for** $y \in Y$

$\lceil R^+ \Leftarrow R^+ \cup \{\langle x, y \rangle\} \rfloor \rfloor \rfloor$

8. Conclusion

Starting from a common high level specification, we have built up a "family tree" of simple transitive closure algorithms.

More recently published algorithms (e.g. [7]) are based on the following concept:

$$x \approx y \Leftrightarrow_{\text{def}} x = y \vee \langle x, y \rangle \in R^+ \wedge \langle y, x \rangle \in R^+.$$

This defines an equivalence relation whose equivalence classes are the strongly connected components of R. As regards R^+, equivalent nodes behave similarly, i.e. more precisely:

$$\text{If } x \approx y \text{ then } \{z \mid \langle x, z \rangle \in R^+\} = \{z \mid \langle y, z \rangle \in R^+\}$$
$$\text{and } \{z \mid \langle z, x \rangle \in R^+\} = \{z \mid \langle z, y \rangle \in R^+\}.$$

These facts can be exploited using Tarjan's algorithm [9] for finding the strongly connected components of R.

As we did not include a description of these facts in our specification we could not hope to derive an algorithm like [7]. It should be worth another exercise to extend the definition and see what algorithms can be derived then.

In our syntheses most of the time we did not manipulate programs. What we did was to transform logical and set theoretical expressions using ordinary mathematical laws. The essence of the algorithms we synthesized is contained in the recursion equations we derived. So we might speak of "program synthesis by manipulating specifications". The advantages of this approach (i.e. staying on a pre-algorithmic level as long as possible) seem to be twofold:

(i) As we are not transforming programs we do not need extra rules. Instead we can make use of any mathematical laws and knowledge.

(ii) There are mathematical laws whose application to a specification will improve the efficiency of the resulting program. As an example we may take the decomposition of a set of pairs (6.9) into a Cartesian product of two sets (6.10) both of which are computed only once by the resulting program. In a similar way it should be possible to describe (and prove more easily!) on the specification level many of the (machine independent) optimizations we apply to make our programs more efficient.

Of course not all the work can be done on the specification level. We realize that in order to obtain the "iterative programs" we illustrated our solutions with, some delicate transformations are needed. In this paper we have concentrated on deriving the shape of an algorithm from a given specification. We think that every transformation step should be done on its appropriate level and that a lot can be achieved on a pre-algorithmic level.

Acknowledgment

I wish to thank Mrs. Hille for her speedy and accurate typing and U. Furbach for his careful reading of several versions of this manuscript.

References

[1] R.M. Burstall and J. Darlington, A transformation system for developing recursive programs, *J ACM* **24**(1) (1977) 44–67.
[2] F.L. Bauer, M. Broy, R. Gnatz, W. Hesse, B. Krieg-Brückner, H. Partsch, P. Pepper and H. Wössner, Towards a wide spectrum language to support program specification and program development, *SIGPLAN Notices* **13** (1978), 15–24.
[3] F.L. Bauer, M. Broy, H. Partsch, P. Pepper and H. Wössner, Systematics of transformation rules, Lecture Notes in Computer Science, **69** (Springer, Berlin, 1979) 273–289.
[4] K. Clark and J. Darlington, Algorithm classification through synthesis, *Comput. J.* **23**(1) (1980) 61–65.
[5] J. Darlington, Application of program transformation to program synthesis, *Proc. IRIA Symposium on Proving and Improving Programs*, Arc-et-Senans, France (1975) 133–144.
[6] J. Darlington, A synthesis of several sort programs, *Acta Informat.* **11** (1978) 1–30.
[7] J. Eve and R. Kurki-Suonio, On computing the transitive closure of a relation, *Acta Informat.* **8** (1977) 303–314.
[8] D. Gries, *Compiler Construction for Digital Computers* (Wiley, New York, 1971).
[9] R. Tarjan, Depth—first search and linear graph algorithms, *SIAM J. Comput.* **1** (1972) 146–160.
[10] S. Warshall, A theorem on Boolean matrices, *J. ACM* (1962) 11–12.